

IN THE SPECIFICATION:

Please replace Algorithm 1 beginning on page ³⁶~~37~~, line ¹²~~16~~ with the following:

Algorithm 1 Simplified Random Line Fit

GenerateNormalizedLineParams(lpParams, ptA, ptB)

{
 $dX = ptA.x - ptB.x$; $dY = ptA.y - ptB.y$; $invNorm = 1.0 / \sqrt{dX*dX + dY*dY}$;
 $lpParams.a = dY * invNorm$;
 $lpParams.b = -dX * invNorm$;
 $lpParams.c = -(lpParams.a * ptA.x + lpParams.b * ptA.y)$;
}

SimpleRandomFit(P, fitLineParameters, distanceThs, strongLineRatio)

{
 stop = false;
 numMaxRejectPts = (1 - strongLineRatio) * size(P);
 trials = 0;
 while (not(stop) AND (trials < maxTrials)) {
 GenerateRandomPair(p, q);
 //Grab points from set P
 ptA = P(p); ptB = P(q);
 lpParams = GenerateNormalizedLineParams(ptA, ptB);
 numRejectPts = 0; ptIndex = 0; setToStart(storePtIndex);
 while ((ptIndex < (size(P) - 1)) AND (numRejectPts < numMaxRejectPts)) {
 if (DistanceLineToPt(lpParams, P(ptIndex)) > distanceThs)
 numRejectPts++;
 else
 ++storePtIndex = ptIndex;
 increment(ptIndex);
 }
 }
}

```

stop = numRejectPts < numMaxRejectPts;
trials++;

```

```

}

```

```

fitLineParameters = FitLineTo(storePtIndex, P);

```

```

}

```

12-16-03 Please replace Algorithm 2 beginning on page ³⁸40, line ²²1 with the following:

Algorithm 2 Random Line Fit

```

RandomLineFit(P, fitLineParameters, distanceThs, strongLineRatio, subsetThsRatio,
              firstPassSizeRatio) {

```

```

    numberOfPoints = size(P);

```

```

    subsetSize = round(firstPassSizeRatio*numberOfPoints);

```

```

    numberOfValidPoints = 0;

```

```

    trials = 0;

```

```

    RandomizePoints(points, numberOfPoints);

```

```

    do {

```

```

        GenerateRandomLine(P, numberOfPoints, &lineParams);

```

```

        Increment(trials);

```

```

        //Check a subset of the total set

```

```

        numberOfValidPoints = CheckSubsetPoints(lineParams, P, distanceThs,
                                                subsetSize, indiceOfValidPoints);

```

```

        if (numberOfValidPoints >= subsetThsRatio*subsetSize) {

```

```

            numberOfValidPoints += CheckRemainingPoints(lineParams, P,
                                                         distanceThs, indiceOfValidPoints);

```

```

            //If a better solution was found

```

```

            if (bestFoundNumberAccepted < numberOfValidPoints) {

```

```

                bestFoundNumberAccepted = numberOfValidPoints;

```

```

                bestLineParams = lineParams;

```

```

                bestIndiceValidPoints = indiceOfValidPoints;
            }
        }
    }
}

```

B2

```

    }
    }
    } while ((numberOfValidPoints < StrongLineRatio*numberOfPoints) AND
              (trials < Max_Trials));

    //if desired a MSE fit can be done, but is not required.
    FitSetPointsMSE(points, bestIndicesValidPoints, &lineParams)
  }

```

Please replace the paragraph beginning on page 1, line 25, which begins, "If the outliers are few..." with the following paragraph:

B3

If the outliers are few and not very far from the ideal line, standard statistical robust techniques can be used to find the best line fitting the set. For more extreme situations more complicated methods exist but tend to be computationally very complex. In some applications, such as image processing, the outliers can be numerous and wide-ranging. In certain problems several lines may be present in a set (see Figures 4 and 610A-C). Usually one is interested in finding the strongest line in the set, which is the line defined by the largest number of points.
